TESYS: E-LEARNING APPLICATION BUILT ON A WEB PLATFORM

Burdescu Dumitru Dan
Software Engineering Department, Faculty of Automation, Computers and Electronics, University of Craiova,
Bvd. Decebal, Nr. 107, 200440, Craiova, Dolj, Romania

Mihăescu Marian Cristian
Software Engineering Department, Faculty of Automation, Computers and Electronics, University of Craiova,
Bvd. Decebal, Nr. 107, 200440, Craiova, Dolj, Romania

Keywords: Web platform, e-learning application.

Abstract: This paper presents a Web platform on which we developed an e-Learning application. The platform represents a web framework in which web applications may be developed. On the platform an e-Learning application is deployed. The e-Learning application brings together professors and students in a collaborative environment managed by secretaries and supervised by administrators. This paper introduces a novel Web application frame based on MVC. This frame separates the business logic from the view and model. Employed technologies, software architecture and software development process improve and ensure the application maintainability, scalability and performance.

1 INTRODUCTION

The main goal of the application is to give students the possibility to download course materials, take tests or sustain final examinations and communicate with all involved parties. To accomplish this, four different roles were defined for the platform: sysadmin, secretary, professor and student.

The main task of sysadmin users is to manage secretaries. A sysadmin user may add or delete secretaries, or change their password. He may also view the actions performed by all other users of the platform. All actions performed by users are logged. In this way the sysadmin may check the activity that takes place on the application. The logging facility has some benefits. An audit may be performed for the application with the logs as witness. Security breaches may also be discovered.

The overall activity of users represents valuable data. This data may be off-line analyzed using machine learning or even data mining techniques such that important conclusions may be obtained regarding the quality of service for the application. The quality of service may have two indicators: the learning proficiency of students and the capability of the application to classify students according to their accumulated knowledge.

A statistics page is also available. It presents the number of users that entered the application, the total number of students, and the number of students with and without activity and other information that gives an overall view on the activity on the application.

Secretary users manage sections, professors, disciplines and students. On any of these a secretary may perform actions like add, delete or update.

These actions will finally set up the application such that professors and students may use it. As conclusion, the secretary manages a list of sections, a list of professors and a list of students. Each discipline is assigned to a section and has as attributes a name, a short name, the year of study and semester when it is studied and the list of professors that teach the discipline which may be maximum three. A student may be enrolled to one or more sections.

The secretaries have also the task to set up the structure of study years for all sections. The structure of a study year is made of a list of periods.

Dumitru Dan B. and Marian Cristian M. (2006).
TESYS: E-LEARNING APPLICATION BUILT ON A WEB PLATFORM.
In *Proceedings of the International Conference on e-Business*, pages 315-318
DOI: 10.5220/0001424803150318

All periods that define the study year are disjunctive in time and are characterized by a name, start date and end date. For each period there are also set up the exams that may be taken and the grants that are needed. The way of defining what the student can do and when proved to be very flexible and easy to understand and use.

The secretaries have the possibility of searching students using different criteria like name, section, year of study or residence. The secretaries have a large set of available reports regarding the student's status. Among them there is a list of students who took all exams, a list of students who requested grant for taking an exam one more time and many other reports specific to secretary work.

The main task of a professor is to manage the assigned disciplines while s discipline is made up of chapters. The professor sets up chapters by specifying the name and the course document. Only students enrolled in a section in which a discipline is studied may download the course document and take tests or examinations. The manner of creating tests and exams is intended to be flexible enough for the professor.

All tests and exams are taken under time constraints. For each chapter the professor sets up a number of seconds necessary to answer questions that chapter. When a test or exam is taken all the seconds are summed thus obtaining a maximal interval of time in which the student has to finish the test. The elapsed and remaining time are managed on server side and presented to the student after each answered question.

The professor has also enough flexibility for creating and editing questions. A question may contain pictures, thus equations, formulas or other graphics may be imbedded into it.

Students may download only course materials for the disciplines that belong to sections where they are enrolled. They can take tests and exams with constraints that were set up by the secretary through the year structure facility.

All users must authenticate through username and password. If the username and password are valid the role of the user is determined and the appropriate interface is presented. The platform assigns a set of actions that the user may perform. Each time a user initiates an action the system checks if that action allowed. This approach ensures security at user's level and makes sure that a student may not perform actions that are assigned to professor, secretary or sysadmin users.

A history of sustained tests is kept for all students. In fact, the taken test or exam is fully saved for later use. That is why a student or a professor may view a taken test or exam as needed. For each question it is presented what the student has checked, which was the correct answer, which was the maximum points that could be obtained from that question and which was the number of obtained points. At the end it is presented the final formula used to compute the grade and the grade itself.

Besides these core functions for on-line testing other ones are implemented or currently under development. A message board is implemented for professors, secretaries and students to ensure peer-to-peer communication. This facility is implemented within the platform such that no other service (e.g. email server) is needed.

The logging facility that is mainly used by sysadmin is transparently implemented for all users (secretaries, professors and students). Whenever one of them performs an action (e.g. a student starts or finishes an exam) that action is recorded for later use.

2 SOFTWARE ARCHITECTURE OF THE PLATFORM

Many issues appear when applications contain a mixture of data access code, business logic code, and presentation code. Such applications are difficult to maintain, because interdependencies between all of the components cause strong ripple effects whenever a change is made anywhere. The Model-View-Controller (MVC for short) design pattern solves these problems by decoupling data access, business logic, and data presentation and user interaction.

The e-learning platform consists of a framework on which a web application may be developed. On server side we choose only open source software that may run on almost all platforms. To achieve this goal Java related technologies are employed.

The architecture of the platform allows development of the e-learning application using MVC architecture. This three-tier model makes the software development process a little more complicated but the advantages of having a web application that produces web pages in a dynamic manner is a worthy accomplishment. The model is represented by DBMS (Data Base Management System) that in our case is represented by MySQL.

The controller, which represents the business logic of the platform, is Java based, being build around Java Servlet Technology. As servlet container Apache Tomcat 5.0 is used.

This architecture of the platform allows development of the e-learning application using MVC architecture. This three-tier model makes the software development process a little more complicated but the advantages of having a web application that produces web pages in a dynamic manner is a worthy accomplishment. The model is represented by DBMS (Data Base Management System) that in our case is represented by MySQL. The controller, which represents the business logic of the platform is Java based, being build around Java Servlet Technology. As servlet container Apache Tomcat 5.0 is used.

The view tier is template based, WebMacro technology being used. WebMacro is also a Java based technology the link between view and controller being done at context level. The separation between business logic and view has great advantages against having them together in the same tier. Once web developer has set up the business logic, web designer can independently work on the view. This decoupling makes development process more productive and safer. One of the biggest disadvantages of having business logic and view together is the lack of modularity that brings problems in application testing and error checking.

Tesys application is an implementation of platform to which new data sources, business logic, and data views may be added.

The business logic of the application uses Java classes. There are four levels of dependency between classes: servlets, actions, managers and beans.

Servlets level has so far two of them: MainServlet and DownloadServlet.

The MainServlet can be seen as the main gate of the application. Its main purpose is to check whether the request may be fulfilled or not from user's role point of view. By default, a user is a visitor and may execute only non-critical actions. The MainServlet redirects visitor users to welcome page where they are supposed to provide a username and a password in order to login. The second level of classes in the dependency tree has the action classes that effectively take care of the requests. The main action classes are Login, Logout, Sysadmin, Secretary, Student and User.

Secretary class codes the actions that may be run by secretary users. There are 64 defined actions that code sections, professors, students, disciplines, exams, grants and feedback management. Ten actions implement secretary's communication with students and professors.

Professor class codes the actions that may be run by professor users. There are 46 defined actions which code disciplines (chapters, test and exam questions), students, grants management. Professors to communicate with students and secretaries use ten actions.

Student class codes the actions that may be run by student users. There are 51 defined actions which code test and exam taking and displaying, maintenance of profile data, communication with secretaries and professors, request of grants to take another examination and fill in feedback forms.

The third level of classes in dependency tree has the manager classes that are extensively used by action classes. Manager classes are in close relationship with the database relations. We have implemented twelve manager classes for the most important relations. A manager class implements all needed operations on a specific relation.

The last level of classes consists of bean classes. These classes are used mainly in manager classes and are in one to one correspondence with the relations from the database. In fact, for each relation there was defined a bean class.

The model is implemented in MySQL and consists of 21 relations. Among the most important ones are: user, role, userrole, usersections, sections, questions, testquestions, examquestions, testresults, examresults, messages, activity.

This database model is used as a completely passive holder of the data that is manipulated entirely by the controller. The model is totally unaware of the existence of either the view or the controller and of its participation in MVC architecture.

The view uses WebMacro templates. Similarly to action classes in Controller, the templates are organized into folders based on the role of the user that will have them displayed.

3 EXPERIMENTAL RESULTS

The platform is currently in use on Windows 2003 Server machine. The setup process consists of two stages. After all needed software is installed (JDK 1.4.2, Apache Tomcat 1.5 and MySQL) the application is deployed. The database structure is created and a sysadmin user is added.

After the setup process finishes the sysadmin user logs in and creates a secretary user. The secretary adds the sections, professors, disciplines and students.

After secretary finishes the setup process professors may start to manage their assigned disciplines. Each professor defines the chapters of each discipline by setting up the name of the chapter and the corresponding course document. After adding a chapter, test and exam questions are also added.

Once the platform is running and all sections, disciplines, course documents and questions are in place, students can also start using the application. This means they can download course materials and take tests and exams.

As a privilege of the sysadmin and professor, they can monitor the activity of every student for each discipline. This monitoring facility provides a good view over the activity of the student. The system keeps tracking when the student logged in, how much time he spent until logging out, which courses were downloaded, how many tests were taken and at what chapters. Table 1 presents the activity of a student regarding a certain discipline.

Table 1: Activity of a student regarding a certain discipline.

Property	Value
Number of downloaded chapters	6
Number of tests from chapter 1 (grades)	10 (9.5, 8, 7.5,6.2,3.3, 9.8,6.5,8.3, 7.1,8.9)
Number of tests from chapter 2 (grades)	9 (8.5, 9, 7.2,5.9,7.3, 9.8,6.5,8.3, 7.1)
Number of tests from chapter 3 (grades)	10 (9.8, 8, 7.9,6.9,7.9, 9.2,7.5,5.3, 8.1,8.9)
Number of tests from chapter 4 (grades)	11 (4.5, 5, 7.8,6.9,5.8, 9.5,6.8,8.7, 7.8,7.6)
Exam grade	8.6

So far, the system was proven to be solid enough with good response time.

4 CONCLUSIONS AND FUTURE WORK

This paper presents how a web application for computer based testing was designed and implemented. The design of the platform is based on MVC model that ensures the independence between the model, the controller and the view.

These choices were very carefully done, only open source software being used. All queries comply SQL Standard so that the DBMS may be changed without interfering the controller and the view part.

The goal of the platform is to create an environment in which students can take tests or exams at different disciplines. Sysadmin, secretary and professors manage the entire infrastructure of the application. The task of the secretary is to manage the general infrastructure consisting of sections, professors, disciplines and students. Professors have to manage their assigned disciplines which means editing questions for testing or examination purposes.

This platform is currently in use and has three sections and at each section four disciplines. Twelve professors are defined and more than 650 students. At all disciplines there are edited almost 2500 questions. In the first month of usage almost 500 tests were taken. In the near future, the expected number of students may be close to 1000.

Recording student's activity under these circumstances provides great information regarding user traffic. After six month of usage there are more than 40,000-recorded actions. Our next concern is to employ different methods of analyzing the traffic in order to create usage patterns and keep the application in good shape.

Other facilities are under continuous development and will give all users the possibility to create a better collaborative environment.

REFERENCES

http://java.sun.com

http://tomcat.apache.org

http://webmacro.org

http://www.mysql.com

http://logging.apache.org/log4j/docs

http://www.junit.org

http://servlets.com/

http://stat257.central.ucv.ro

http://www.cncsis.ro

G. E. Krasner and S. T. Pope, "A cookbook for using the model-view-controller user interface paradigm in smalltalk-80", JOOP, Aug./Sept. 1988.

E. Gamma, R. Helm, R. Johnson, and J. Vlissides. "Design Patterns - Elements of Reusable Object-Oriented Software", Addison Wesley, 1995.

P. Coad and E. Yourdon, "Object-oriented Design" Yourdon Press (Prentice Hall), New Jersey, 1991.